Journal of Engineering and Applied Sciences 12 (Special Issue 8): 8355-8359, 2017
ISSN: 1816-949X

Assessing the Efficiency of Scoring Rubrics Assisted Reading Approach in the Review of Software Requirements Work Products

Emmanuel O.C. Mkpojiogu and Nor Laily Hashim
School of Computing, Universiti Malaysia, 06010 Sintok, Malaysia

Abstract: The efficiencies of the various reading techniques are at varying degrees. Earlier researcher on the efficiency of reading techniques reveal that perspective-based reading is more efficient than the other traditional reading techniques: ad-hoc and checklist. This aside the studies reviewed in this study, placed more emphasis on the effectiveness of reading techniques. The efficiency of these techniques received little consideration. This study assesses the efficiency of Scoring Rubric Assisted Reading (SRAR). The technique uses a checklist-like mechanism to assist readers in finding defects in software requirements documents and artifacts. No prior research has been carried out on assessing the efficiency of SRAR in the past. Reading time and reading efforts were used as measurement metrics. The research approach employed involved the following steps: using SRAR to read artifacts, detecting defects, measuring the time used in the reading process per artifact and computing the effort expended per artifact for each round/iteration of review. Preliminary results show the efficiency of this mechanism and technique.

Key words: Efficiency, scoring rubric assisted reading, SRAR, requirements work product, review, artifact

INTRODUCTION

One of the most common and costly mistakes made in software projects nowadays is the deferring of the activity of detecting and correcting software faults and defects until too late in the project's lifecycle. This results from the principle that the costs of a defect increase significantly as the project's lifecycle progresses if the the defect is discovered and corrected (Boehm, 1981). Consequently, if lower costs and higher quality are the goals, defect detection and removal should not be limited to phases at the end of the software development process but should be part of each individual development phase right from the start. This thus requires the use of other techniques, methods and tools like inspection and reviews in addition to software testing (Laitenberger, 2000). Software inspection is an approach that allows the early cost-effective detection and removal of defects immediately after software documents and artifacts are created. In addition with this method, rework cost are reduced immensely, since, defects are typically found directly after they are introduced. Also, early defect detection and removal improve the predictability of software projects and assists project managers to stay within project schedule, since, problems are promptly unveiled throughout the early construction phases and costly rework cycles at the end of the software construction or maintenance project are avoided (Laitenberger, 2000). The defect detection phase can be considered the core of an inspection. The main goal of the defect detection phase is to scrutinize and examine a software document to identify defects and faults. (Laitenberger, 2000). Inspection is a regularly employed strategy in verification. During inspection, inspectors read the software artifacts to determine whether quality requirements such as correctness, consistency, testability or maintainability have been fulfilled. Reading, thus implies the systematic examination of a document to extract, gain and understand certain information about the inspected software (Laitenberger, 2000).

This study placed more emphasis on the effectiveness of reading techniques. The efficiency of these techniques received little attention. This study therefore seeks to investigate the efficiency of Scoring Rubric Assisted Reading (SRAR). However, the study will not compare existing techniques with SRAR as this study is an exploratory one.

Literature review: Software reading/review/inspection is a way of developing high quality software. It provides assurance on the quality of a software work product (Berling and Runeson, 2003). The quality of work products such as the software requirements specification, etc., is the key to the success of software development. The inspection for verification and validation of software documents are widely practiced but the approaches used

Corresponding Author: Nor Laily Hashim, School of Computing, Universiti Malaysia, 06010 Sintok, Malaysia

are rather ad-hoc or better still, checklist-based but largely depending on the knowledge and skill of the review personnel (Saito *et al*., 2014). There are a number of reading techniques, these are: ad-hoc reading, checklist reading, defect-based reading and Perspective-Based Reading (PBR) (Basili and Selby, 1987). Others include: function-point reading and reading by stepwise abstraction (Laitenberger, 2000).

Cheng and Jeffery (1996) carried out an experiment to investigate the effectiveness of this approach compared to an ad-hoc approach. The experimental results reveal that on average, inspectors following the ad-hoc approach found more defects than inspectors following the function-point scenarios. However, it seems that experience is a confounding factor that biased the results of the experiment. Function point analysis requires the inspector to check the documents according to the function point items (Laitenberger, 2000).

Also, Laitenberger presented a report based on three experiments and indicated that perspective-based reading improved best the performance of reviewers that have medium skills and less for those with very little or very high skills. This observation revealed that very experienced and skilled reviewers do not depend on the use of perspective-based reading while reading documents. It also showed that little skilled reviewers find it hard to cope, since, scenarios use individual abilities for specific model building and role (Laitenberger *et al*., 2001). Laitenberger further asserts that PBR improves the cost-effectiveness (that is efficiency) of reading in comparison with the other more traditional reading techniques (Laitenberger, 2000).

More so, Porter and Votta (1994) compared scenario-based reading, checklist-based reading and ad-hoc reading in requirements specification inspection. They found no significant difference between ad-hoc and checklist-based reading techniques. The experiment was replicated by Porter *et al*. (1995). Their study evaluated and compared defect-based reading, ad-hoc reading and checklist-based reading. They found that: the defect-based readers performed better than ad-hoc and checklist readers; the defect-based reading procedures assisted reviewers to focus on specific defect classes. From the forgoing comparisons, it is evident that the efficacy of the reading techniques grows in the following order; ad-hoc; checklist and scenario-based reading techniques. The list grows from the most rudimentary and less defined to the more defined and complex techniques. The SRAR reading technique is conceptually somewhere in the middle. SRAR is defined and yet simple to use (Mkpojiogu and Hashim, 2017). Earlier works on the efficiency of reading techniques reveals that perspective-based reading is more efficient that the other traditional reading techniques: ad-hoc and checklist (Laitenberger, 2000).

Worthy of mention is the place of efficiency of reading techniques. ISO (2011) prescribed a quality model with five characteristic as quality in use, namely: effectiveness, efficiency, freedom from risk, satisfaction and context coverage. It also outlined eight other characteristics as internal and external qualities, namely: functionality, suitability, compatibility, security, reliability, usability, performance efficiency, maintainability and portability. This study however, employs efficiency which is also closely related to usability (IEEE Std. 1998; ISO., 2011). Efficiency, a quality measure that defines the extent at which resources are expended in the course of achieving a given goal is very important in the use of software reading techniques. As software reading is an essential part of software quality assurance, the quality of requirements work products and of software is thus reliant on how well these work products are read and reviewed. The level of resources that is expended in the process of reading a piece of requirements document while reading the documents for defects define its efficiency. Efficiency of a reading technique thus is the extent at which the reading technique can detect defects and errors in a software requirements work product with minimal resources (cost-effectiveness). Among resources that are expended in software reading and defect detection include: time, effort and finance, etc. These resources are related in use. When so much resource is spent in the reading and defect detection process it implies that the reading technique and/or mechanism have poor efficiency. However, if fewer resources are used in the reading and defect detecting exercise, then the reading approach is said to have good efficiency. Reading efficiency combines effort and effectiveness and is defined as the number of defects detected per unit time (Abrahamsson, 2007).

MATERIALS AND METHODS

SRAR technique was employed to support the detection of defects in requirements work documents and products. It was also used to evaluate the quality of the requirements products. In the course of using the SRAR technique for the reading of requirements work products and in the detection of defects, the efficiency of the SRAR technique was also measured and evaluated. A scoring rubric is a two-dimensional Likert-like instrument. The columns define a 4-point Likert-type rating scale (for instance, not acceptable, below expectations, meet expectations, exceeds expectations); a 15th column is

added for "not applicable". The rows consist of the attributes of the given work products/artifact. In some cases, the attributes are further defined into criteria. In addition, the cells created by the intersection of the rows and columns represent a clear description of the work product's attributes with regard to the corresponding rating scale. Each attribute is scored and the scores of all attributes are totaled to form a total score for the given work product. The scores of the rubrics are performance metrics. Some metrics were used to assess the efficiency of the SRAR technique. They are inter alia: review/reading time and reading effort. These metrics assesses efficiency of SRAR technique (they assess the cost put into the reading and defect detecting process). Efficiency determines the amount of resources expended in order to achieve success with the SRAR technique. These metrics also describe the efficacy of SRAR which implies how powerful the technique is. They also describe the usability of the technique in terms of how usable (easy to use) they are. In this study, review/reading time and reading effort were used to measure the efficiency of SRAR. Review time is the time expended in detecting defect while reading and reviewing a requirements work product. Effort is the number of defects detected in a requirements work product per unit time (in this case, defect detected per minute of reading of the artifact). Prior research used these metrics in measuring reading efficiency (Abrahamsson, 2007). In this study, the measure of time was in minutes.

This study was conducted in School of Computing, Universiti Utara Malaysia. The Software Engineering sub-department has in recent years been using scoring rubrics in the evaluation of the quality of student's software/requirements documents and models. But the instrument's efficiency has not been empirically assessed and validated. This study is the first attempt to offer an empirical assessment of the efficiency of the tool and reading mechanism. This study was part of the study carried out during the development of an e-Health awareness system (Hussain and Mkpojiogu, 2016; Hussain *et al*., 2015; 2016). After the requirements documents and models were produced they were read and reviewed by 4 experts (Senior Lecturers) in two iterations who detected defects in the documents. Each reviewer read and reviewed the requirements documents and models separately and independently. At the end of each of the two reading iterations, the requirements work products were refined and the defects removed. The following research question guided the study: how efficient is the SRAR technique? Descriptive statistics was employed in answering the research question. The following metrics were used in the study: review/reading time and reading effort.

RESULTS AND DISCUSSION

Review/reading time: Table 1 shows the average time expended in the review and reading process in the second round of the inspection is lower than that of the first. This indicates that lesser resource (in terms of time) was spent in the second inspection iteration when compared to the first and it also shows that the reading technique has some level of efficiency. In this metrics, a percentage reduction in review time is the expected outcome. The study shows a 100% success rate in the reduction of reading/review time as there was none of the work documents that the reviewers did not achieve a reduction in review time in the second round of the reading effort. The percentage reduction for the various requirements work products used in this study are as follows: vision and scope document 56%, use case description 50%, software requirements specification 65%, test plan 50%, test cases 25%, combined requirements models 60%, average reading time for all textual documents combined 50%, average reading time for all textual documents and requirements models put together 52%. From this presentation, the work products with the least percentage reduction in time are the test cases with a percentage reduction of 25%. All other documents averaged from 50-65%. This indicates a very good efficiency for the SRAR reading/review technique. SRAR is a cost-effective and efficient reading technique.

Reading effort: On the average, the effort that is defects detected/minute put into finding defects in the second round of the inspection is generally, lower than that put into the first round (Table 2). This is an indication of the efficiency and cost-effectiveness of SRAR as lesser work is done in round two when compared to round one there are reduced defects to search for in round two in comparison to those searched for in round one of the reading process. This again indicates efficacy as well as the efficiency of the SRAR technique. Reduction in reading efforts is the expected outcome for an efficient reading technique. Except for one work product, i.e., vision and scope documents all other documents achieved a reduction in review efforts. Both use case description and software requirements specification, achieved a 100% reduction in review efforts. Other work documents with their corresponding percentage reduction in reading efforts are as follows: test plan 14%, test cases 35%, requirements models combined 58%. The average reading effort for all textual documents combined 57% and the average reading efforts for all textual documents and requirements models combined 60%. Therefore, using

Table 1: Average reading time for two rounds of reading: overall average time: 7.79 min/artifact

Work product	Time (min) (round 1)	Time (min) (round 2)	Difference	Reduction in review time (%)	Improvement
Vision and scope doc.	9.00	4.00	5.00	56	√
Use case description	10.00	5.00	5.00	50	√
Software requests spec	11.50	4.00	7.50	65	√
Test plan	10.00	5.00	5.00	50	√
Test cases	10.00	7.50	2.50	25	√
Reqs models	12.50	5.00	7.70	60	√
Ave. reading time for docs	10.10	5.10	5.00	50	√
Ave. reading time for docs and requests models	10.50	5.08	5.42	52	√

Table 2: Average effort for two rounds of reading: overall effort: 0.29 defects/minute

Work product	Effort (defs/min) (round 1)	Effort (defs/min) (round 2)	Difference	Reduction in effort (%)	Improvement
Vision and scope doc.	0.11	0.13	-0.02	-18	X
Use case description	0.40	0.00	0.40	100	√
Software requests spec	0.30	0.00	0.30	100	√
Test plan	0.35	0.30	0.05	14	√
Test cases	0.20	0.13	0.07	35	√
Requests models	0.72	0.30	0.42	58	√
Ave. reading effort for docs	0.28	0.12	0.16	57	√
Ave. reading effort for docs and requests models	0.37	0.15	0.22	60	√

SRAR can achieve efficient defect detection and work product and software artifact's quality improvement. Two metrics were employed to assess the efficiency of SRAR: reading time and review effort. In all the metrics, it is observed that after round two of the reading and review there was observable reduction in reading and defect detecting cost and corresponding improvement in the work product's quality levels as evidenced in all the metrics used. The time used for the reading of requirements models reduced by 50% on the average and that of requirement models plus other text-based documents reduced by 52%. On the average, the reading time per work products was 7.79 min, signifying efficiency as the readers were able to sufficiently detect defects per document in <8 min of reading. In addition, there was a considerable reduction is review efforts. The percentage reduction in effort for requirements models and the combination of requirements models and textual documents were 57 and 60%, respectively. This reduction in cost (effort) from round one to two of the reading and review indicates and reveals the efficiency of the reading/review instrument and technique.

CONCLUSION

Software has become more pervasive in today's society, hence, the importance of software quality is constantly increasing. Software inspections help to improve the quality of software. SRAR has promised to be an efficient reading technique that overcomes the shortcomings and inadequacies of the checklist and ad-hoc reading techniques. It is a very simple and defined reading approach that not only provides guides to reviewers but is also convenient, usable and consistent. It has efficacy in assisting reviewers/readers detect defects in software/requirements artifacts. More so, it can be utilized to steadily watch the observed progress in the quality improvements of work products. This study nonetheless was an exploratory one and was thus; limited in the sense that the efficiency of SRAR approach was not empirically compared with that of checklist or any other reading technique but the efficiency was adjudged from the defect defection capacity and quality improvement potentials of SRAR as seen in the study.

RECOMMENDATION

Future research will concentrate on evaluating and comparing the efficiency of SRAR with those of other reading techniques using both professional and non-professional users/readers.

REFERENCES

Abrahamsson, J.M.P., 2007. Product-focused software process improvement. Proceedings of the 8th International Conference on Product-Focused Software Process Improvement (PROFES 2007), July 2-4, 2007, Springer, Riga, Latvia, pp: 175-187.

Basili, V.R. and R.W. Selby, 1987. Comparing the effectiveness of software testing strategies. IEEE. Trans. Software Eng., 13: 1278-1296.

Berling, T. and P. Runeson, 2003. Evaluation of a perspective based review method applied in an industrial setting. IEEE. Proc. Software, 150: 177-184.

Boehm, B.W., 1981. Software Engineering Economics. Prentice-Hall, Englewood Cliffs, New Jersey, ISBN: 9780138221225, Pages: 767.

Cheng, B. and R. Jeffery, 1996. Comparing inspection strategies for software requirement specifications. Proceedings of the Conference on Australian Software Engineering, July 14-18, 1996, IEEE, Melbourne, Victoria, Australia, ISBN:0-8186-7635-3, pp: 203-211.

Hussain, A. and E.O.C. Mkpojiogu, 2016. An application of Kano method in the elicitation of stakeholder satisfying requirements for an E-ebola awareness system. Intl. J. Syst. Appl. Eng. Dev., 10: 169-178.

Hussain, A., E.O.C. Mkpojiogu and F.M. Kamal, 2015. Eliciting user satisfying requirements for an e-health awareness system using kano model. Proceedings of the 14th WSEAS International Conference on Computer and Conputational Science (ACACOS'15), April 23-26, 2015, WSEAS Press, Kuala Lumpur, Malaysia, ISBN:978-1-61804-297-2, pp: 156-165.

Hussain, A., E.O.C. Mkpojiogu and M.N.M. Nawi, 2016. Requirements model for an E-health awareness portal. Proceedings of the International Conference on Applied Science and Technology (ICAST'16) Vol. 1761, April 11-13, 2016, AIP, Kedah, Malaysia, pp: 020048-020048.

IEEE Std 830, 1998. IEEE recommended practice for software requirements speci?cations. Software Engineering Standards Committee of the IEEE Computer Society, pp: 11.

ISO., 2011. IEC25010: 2011 Systems and software engineering-systems and software quality requirements and evaluation (SQuaRE)-system and software quality models. International Organization for Standardization, Geneva, Switzerland.

Laitenberger, O., 2000. Cost-effective detection of software defects through perspective-based inspections. Ph.D Thesis, University of Kaiserslautern, Kaiserslautern, Germany.

Laitenberger, O., E.K. Emam and T.G. Harbich, 2001. An internally replicated quasi-experimental comparison of checklist and perspective based reading of code documents. IEEE. Trans. Software Eng., 27: 387-421.

Mkpojiogu, E.O.C. and N.L. Hashim, 2017. Improving the quality of software requirements work products using scoring rubrics assisted reading. Proceedings of the 6th International Conference on Computing and Informatics (ICOCI'17), April 25-27, 2017, Universiti Utara Malaysia, Kuala Lumpur, Malaysia, pp: 656-662.

Porter, A.A. and L.G. Votta, 1994. An experiment to assess different defect detection methods for software requirements inspections. Proceedings of the 16th International Conference on Software Engineering, May 16-21, 1994, IEEE Computer Society Press, Los Alamitos, California, USA., ISBN:0-8186-5855-X, pp: March 27, 2018103-112.

Porter, A.A., L.G. Votta and V.R. Basili, 1995. Comparing detection methods for software requirements inspections: A replicated experiment. IEEE. Trans. Software Eng., 21: 563-575.

Saito, S., M. Takeuchi, S. Yamada and M. Aoyama, 2014. RISDM: A requirements inspection systems design methodology: Perspective-based design of the pragmatic quality model and question set to SRS. Proceedings of the 2014 IEEE 22nd International Conference on Requirements Engineering (RE), August 25-29, 2014, IEEE, Karlskrona, Sweden, ISBN:978-1-4799-3033-3, pp: 223-232.